REMARKS

I. INTRODUCTION

Claims 1 and 45-49 have been amended to more particularly point out and distinctly claim the subject matter of the present invention. No new matter has been added. Claims 1-49 remain pending in the present application. In view of the above amendments and the following remarks, it is respectfully submitted that all of the presently pending claims are allowable.

II. THE 35 U.S.C. § 112 REJECTIONS SHOULD BE WITHDRAWN

The Examiner rejected claim 1-49 under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which the applicant regards as the invention. (*2/10/05 Office Action*, ¶¶ 7,11,12, pages 2-3.) Claims 1 and 45-49 have been amended to provide a sufficient antecedent basis for each claim term. As amended, these claims satisfy the requirements of 35 U.S.C. 112, second paragraph. Therefore, it is respectfully requested that the rejection of these claims and any claims depending therefrom (claims 2-44) be withdrawn.

III. THE 35 U.S.C. § 102(b) REJECTIONS SHOULD BE WITHDRAWN

The Examiner rejected claims 1-9, 11-15, 18-19, 22-44, 46, 48, and 49 under 35 U.S.C. 102(b) as being anticipated by U.S. Patent No. 5,689,712 to Heisch (hereinafter "Heisch"). (*2/10/05 Office Action*, ¶¶ 8,14,15, pages 2-16). Heisch describes a method and system for optimizing programs, having memory references, at the object code level. (*See*

PAGE 16/23 * RCVD AT 6/8/2005 4:40:40 PM [Eastern Daylight Time] * SVR:USPTO-EFXRF-1/5 * DNIS:8729306 * CSID:212 619 0276 * DURATION (mm-ss):07-56

Heisch, Abstract). This optimization is performed by means of a postprocessor which profiles a program at the object code level, analyzes memory reference patterns, and then restructures the memory allocation in the program. (*See Heisch,* Col. 2, ll. 15-20). Specifically, an instrumentation code at the end of a program contains a main loop starting with a search of the program for a load or store instruction. (*See Heisch,* Col. 4, ll. 7-9). Once located, the postprocessor replaces the instruction with an unconditional branch to the start of the instrumentation code. (*See Heisch,* Col. 4, ll. 5-14). The postprocessor then appends the load/store instruction to the end of the instrumentation code, and thereafter appends an unconditional branch back to the original code path immediately following the replaced load/store instruction to preserve the original program behavior. (*See Heisch,* Col. 4, ll. 15-20). This process reiterates until each load/store instruction of the program is instrumented. (*See Heisch,* Col. 4, ll. 22-29).

Claim 1 of the present application describes a method of instrumenting software code to improve runtime execution. In the claimed method, a desired instruction within a software code is identified and replaced "with a program flow change instruction *directing execution to a buffer.*" An example of such a buffer is an exception vector table, wherein subsequent instructions in the vector table may execute the desired instruction. (*See Specification,* p. 7, ll. 46-51). In contrast, Heisch replaces the instruction with a branch to the beginning of the instrumentation code located at the end of the program. The portion of Heisch cited by the Examiner mentions a buffer, but this is not relevant to the present invention because the buffer in Heisch is merely used to store addresses of memory references to facilitate tracing

PAGE 17/23 * RCVD AT 6/8/2005 4:40:40 PM [Eastern Daylight Time] * SVR:USPTO-EFXRF-1/5 * DNIS:8729306 * CSID:212 619 0276 * DURATION (mm-ss):07-56

an instrumented program. (*See Heisch*, Col. 4, ll. 31-45). It does not actually execute any routine or instruction.

Additionally, the invention of the present application differs from Heisch in that inserted into the above described buffer is also a "routine *having an output instruction*." This output instruction enables both user selectable functions and code executing on a bus of the target to be displayed (e.g., via a GUI). (*See Specification*, p. 18). Heisch, however, fails to disclose the incorporation of any such output instruction.

Furthermore, optimization of a program code generally occurs when the program is made to run more quickly or to take up less space. The present invention as claim is aimed at optimizing software code by changing the behavior "*of the runtime execution.*" In contrast, the method of optimization disclosed in Heisch occurs through restructuring of memory allocation, and therefore reducing the requisite space. Thus, although both Heisch and the present application may regard optimization of software code, a separate and distinct form of optimization is achieved by each.

In view of the above, it is respectfully submitted that the rejection of claim 1 be withdrawn. Because claims 2-9, 11-15, 18-19, and 22-44 depend from and therefore include all the limitations of claim 1, Applicants submit that these claims are also allowable.

The Examiner rejected claims 46, 48, and 49 under the same reasoning used in the rejection of claim 1. (*See 4/25/05 Office Action*, ¶ 14, p. 16). These claims recite substantially the same limitations as claim 1. Specifically, claim 46 recites "an instruction replacement module configured to replace the desired instruction with *a program flow change instruction directing execution to a buffer*, wherein the program flow change instruction is configured to

PAGE 18/23 * RCVD AT 6/8/2005 4:40:40 PM [Eastern Daylight Time] * SVR:USPTO-EFXRF-1/5 * DNIS:8729306 * CSID:212 619 0276 * DURATION (mm-ss):07-56

change the behavior, relative to that of the desired instruction, *of the run time execution* of the software code." Claim 48 recites a computer usable medium having "computer readable program code for replacing the desired instruction with a program flow change instruction *directing execution to a buffer*, wherein the program flow change instruction is configured to change the behavior... *of the run time execution* of the software code." Claim 49 recites "computer readable program code for replacing the desired instruction with a program flow change instruction *directing execution to a buffer*, wherein the program flow change instruction is configured to change the behavior... of the run time execution of the software code." Therefore, Applicants submit that these claims are also allowable for the same reasons discussed above with respect to claim 1, and respectfully request that the rejections of these claims be withdrawn.

IV. THE 35 U.S.C. § 103(a) REJECTIONS SHOULD BE WITHDRAWN

The Examiner rejected claims 20-21, 45, and 47 under 35 U.S.C. 103(a) as being unpatentable over Heisch in view of U.S. Patent No. 5,664,191 to Davidson et al. (hereinafter "Davidson"). (*2/10/05 Office Action*, ¶¶ 9,15,16, pages 2, 16-18.) In support of these rejections, the Examiner incorporated the §102(b) rejection of claim 1 as unpatentable over Heisch. (*See id.*). Acknowledging, however, that Heisch fails to disclose "wherein the searching comprises using debug information to identify the desired instruction," the Examiner introduces Davidson to cure this defect.

Similar to Heisch, Davidson describes a method and system for optimizing a computer program by improving the locality of memory references. (*See Davidson,* Abstract). Specifically, an optimal placement order for basic blocks within a computer program is

PAGE 19/23 * RCVD AT 6/8/2005 4:40:40 PM [Eastern Daylight Time] * SVR:USPTO-EFXRF-1/5 * DNIS:8729306 * CSID:212 619 0276 * DURATION (mm-ss):07-56

determined by analyzing the program in machine code format to identify the basic blocks, assigning a placement order to each basic block depending on the number of times it is executed, and reordering the blocks accordingly. (*See Davidson*, Col. 2, ll. 26-45). The memory required to execute the program is thereby reduced. (*See id.*).

Davidson fails to cure the defects of Heisch discussed hereinabove with respect to the §102(b) rejection of claim 1. Accordingly, the Applicants respectfully submit that Heisch and Davidson, either alone or in combination, neither teach nor suggest identifying and replacing an instruction "with a program flow change instruction *directing execution to a buffer*, wherein the program flow change instruction is configured to change the behavior... *of the run time execution*" as recited in claims 20, 21, 45, and 47. Thus, the Applicants respectfully request that the Examiner withdraw the rejections of these claims.

PAGE 20/23 * RCVD AT 6/8/2005 4:40:40 PM [Eastern Daylight Time] * SVR:USPTO-EFXRF-1/5 * DNIS:8729306 * CSID:212 619 0276 * DURATION (mm-ss):07-56

CONCLUSION

In view of the amendments and remarks submitted above, the Applicant respectfully submits that the present case is in condition for allowance. All issues raised by the Examiner have been addressed, and a favorable action on the merits is thus earnestly requested.

Respectfully submitted,

Dated: June 8, 2005

By: _____
Michael J. Marcin (Reg. No. 48,198)

Fay Kaplun & Marcin, LLP
150 Broadway, Suite 702
New York, NY 10038
(212) 619-6000 (phone)
(212) 619-0276 (facsimile)

PAGE 21/23 * RCVD AT 6/8/2005 4:40:40 PM [Eastern Daylight Time] * SVR:USPTO-EFXRF-1/5 * DNIS:8729306 * CSID:212 619 0276 * DURATION (mm-ss):07-56